ReadMe File for FAST v8.08.00c-bjj

Bonnie Jonkman, Jason Jonkman
National Renewable Energy Laboratory
June 30, 2014

Introduction

This document is designed to guide you through some of the changes that the FAST wind turbine computer-aided engineering (CAE) tool is undergoing. FAST v8.08.00c-bjj is the second public release of FAST under the new modularization framework developed at NREL. The architecture of FAST v8 is entirely different from FAST v7.02.00d-bjj. These differences are highlighted in Figure 1.

The modules of FAST (AeroDyn, HydroDyn, etc.) correspond to different physical domains of the coupled aero-hydro-servo-elastic solution, most of which are separated by spatial boundaries. Figure 2 shows the control volumes associated with each module for fixed-bottom offshore wind turbines. For land-based wind turbines, the HydroDyn hydrodynamics module would not be used and the SubDyn multi-member substructure structural-dynamics module is optional. Figure 3 shows the control volumes associated with each module for floating offshore wind turbines.

While FAST v8 has many features not found in FAST v7, several features of FAST v7.02.00d-bjj have not yet been added to FAST v8, so we will continue to support both versions of the software (FAST v7 and FAST v8) until FAST v8 is deemed a suitable replacement. Table 1 summarizes the different features available in each version.

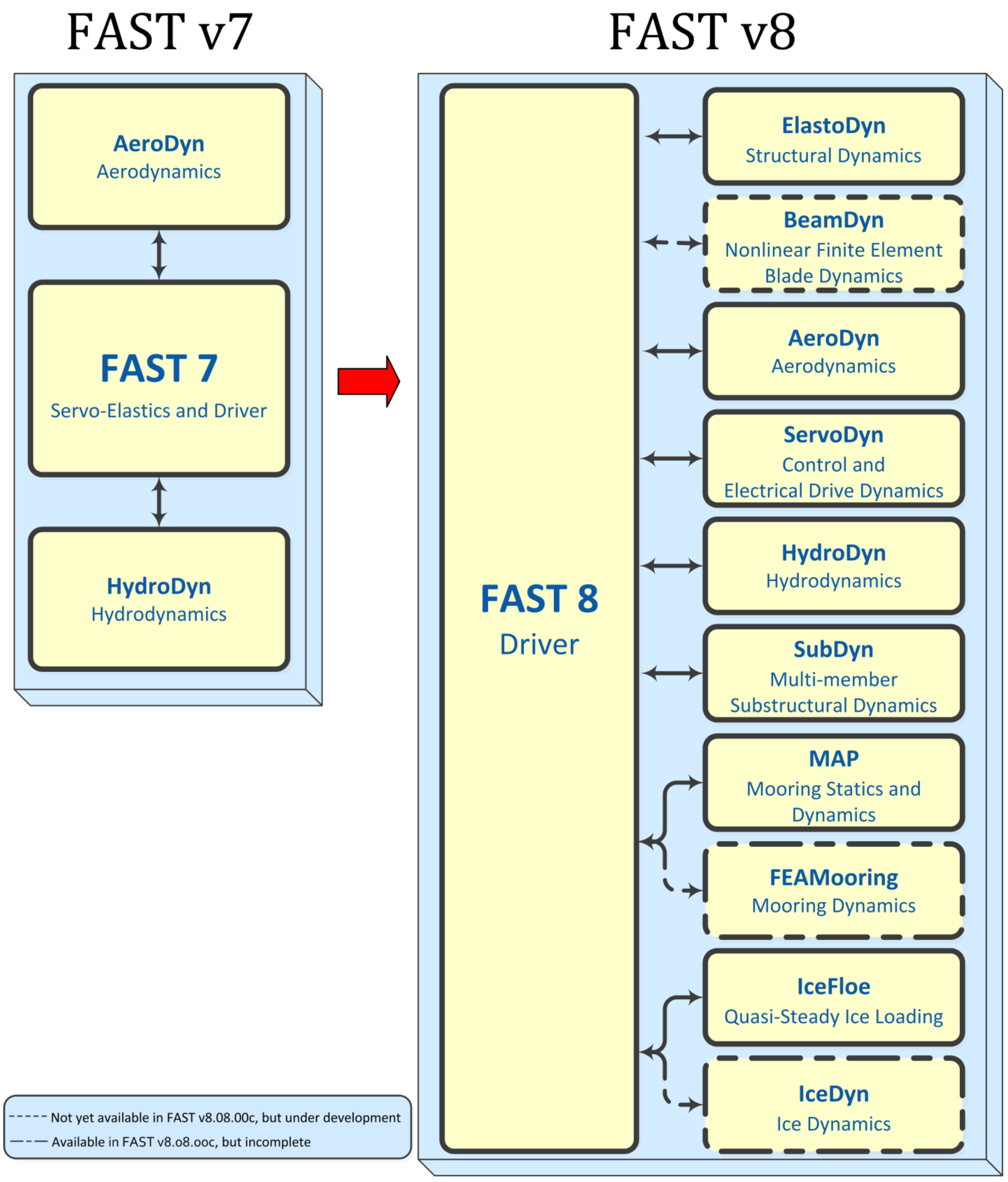

FAST v7
FAST v8
AeroDyn
Aerodynamics
FAST 7
Servo-Elastics and Driver
HydroDyn
Hydrodynamics
FAST 8
Driver
ElastoDyn
Structural Dynamics
BeamDyn
Nonlinear Finite Element Blade Dynamics
AeroDyn
Aerodynamics
ServoDyn
Control and Electrical Drive Dynamics
HydroDyn
Hydrodynamics
SubDyn
Multi-member Substructural Dynamics
MAP
Mooring Statics and Dynamics
FEAMooring
Mooring Dynamics
IceFloe
Quasi-Steady Ice Loading
IceDyn
Ice Dynamics
Not yet available in FAST v8.08.00c, but under development
Available in FAST v8.08.00c, but incomplete

Figure 1: Architectural comparison of FAST 7 and FAST 8

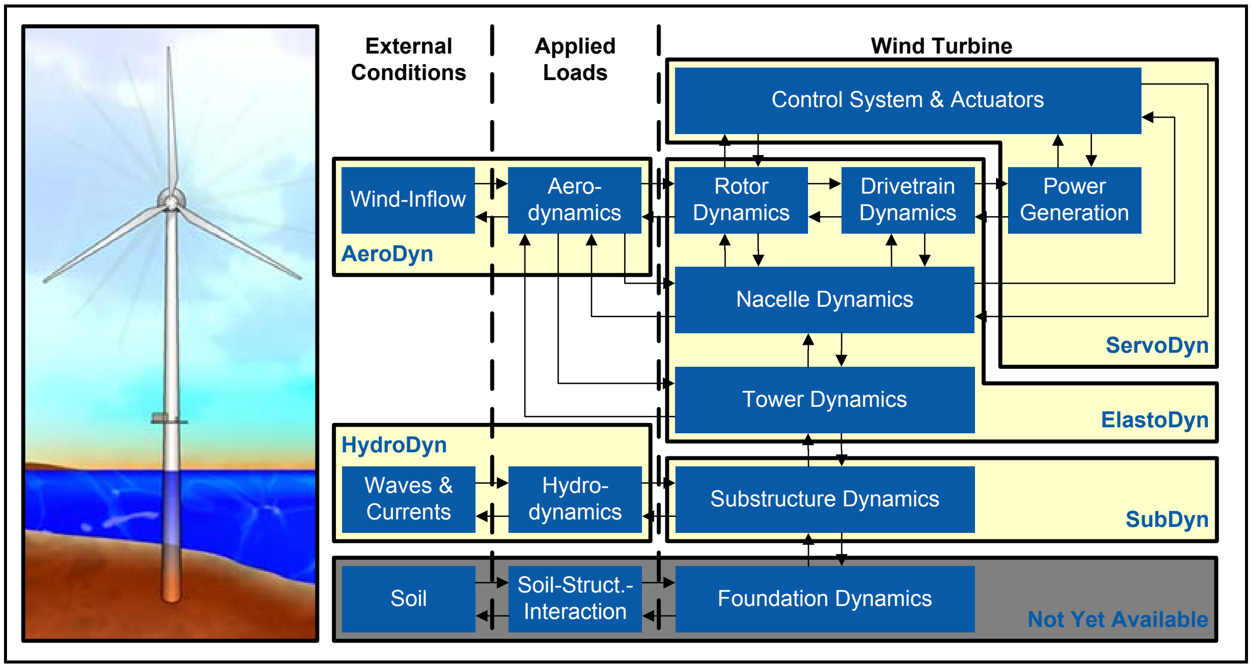


Figure 2: FAST control volumes for fixed-bottom systems (surface ice not shown)

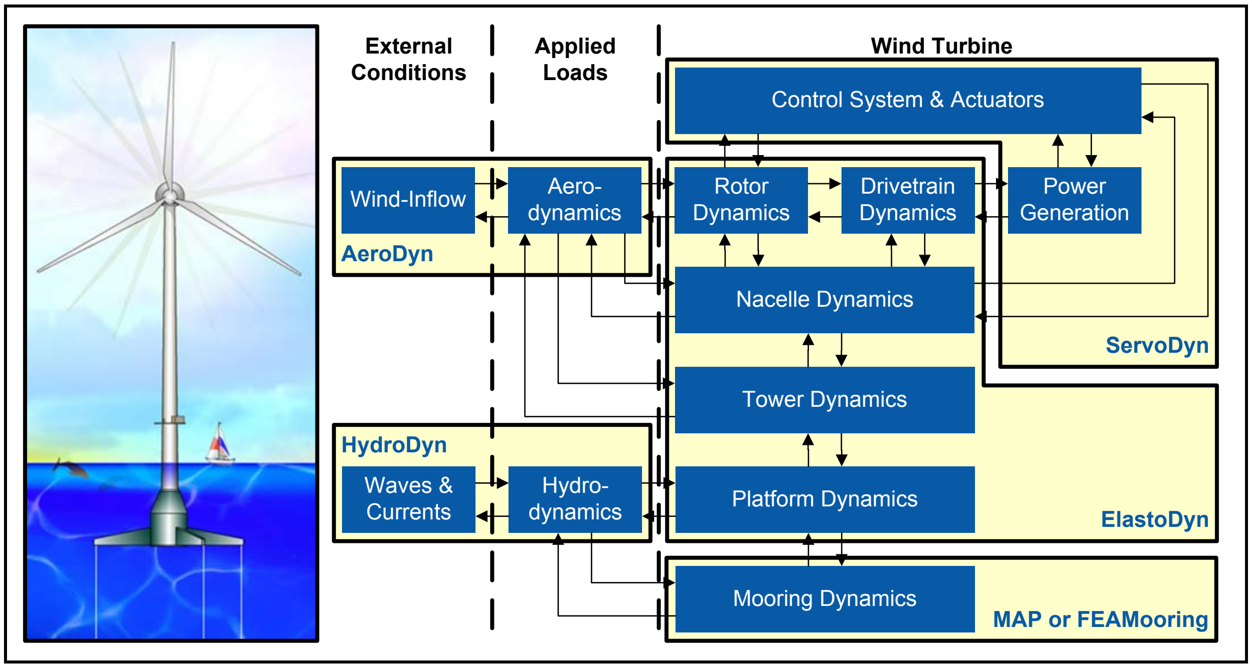


Figure 3: FAST control volumes for floating systems

Table 1: Comparison of features between FAST v7 and v8

Aerodynamics (AeroDyn and InflowWind)

FAST Features	v7.02	v8.08
• Quasi-steady or dynamic wake	✓	✓
• Steady or unsteady airfoil aerodynamics	✓	✓
• Tower shadow for downwind rotors	✓	✓
• Tower influence for upwind rotors	✓	✓
• Tower drag loading		✓
• Tail-fin aerodynamic loading	✓	
• "Hub-height", TurbSim, and GH Bladed wind file formats	✓	✓
• Other wind formats	✓	
• Aeroacoustics (noise)	✓	

Hydrodynamics (HydroDyn, IceFloe, and IceDyn)

FAST Features	v7.02	v8.08
• Linear regular or irregular waves	✓	✓
• White-noise waves		✓
• Wave directional spreading		✓
• Wave stretching	✓	
• Externally generated wave data	✓	
• Sea current	✓	✓
• Strip theory for central member	✓	✓
• Strip theory for multiple intersecting members		✓
• Distributed static buoyancy		✓
• Concentrated loads on member ends		✓
• Support for inclined and tapered members		✓
• Support for flooded and ballasted members		✓
• Support for marine growth		✓
• First-order potential flow (from WAMIT)	✓	✓
• Radiation "memory effect" captured through time-domain convolution	✓	✓
• Radiation "memory effect" captured through linear state-space form		✓
• Quasi-steady and dynamic surface-ice loading		✓

Control and Electrical System (Servo) Dynamics (ServoDyn)

FAST Features	v7.02	v8.08
• Blade-pitch control	✓	✓
• Override pitch maneuvers	✓	✓
• Generator models	✓	✓
• Torque control	✓	✓
• High-speed shaft brake	✓	
• Nacelle-yaw control	✓	✓
• Override yaw maneuvers	✓	✓
• Blade-tip brakes	✓	
• GH Bladed DLL interface*	✓	✓
• Simulink interface	✓	
• LabVIEW interface	✓	

Structural Dynamics (ElastoDyn, SubDyn, MAP, and FEAMooring)

FAST Features	v7.02	v8.08
• Blade-bending DOFs	✓	✓
• Rotor-teeter DOF	✓	✓
• Generator azimuth and drivetrain torsion DOFs	✓	✓
• Nacelle-yaw DOF	✓	✓
• Tower-bending DOFs	✓	✓
• Rigid-body platform DOFs	✓	✓
• Furling DOFs	✓	
• Fixed-bottom multi-member substructure DOFs:		✓
• Gravitational loading	✓	✓
• Gearbox friction	✓	
• System of independent mooring lines solved quasi-statically	✓	✓
• System of multi-segmented mooring lines solved quasi-statically		✓
• Mooring dynamics		✓
• Earthquake excitation	✓	

* This option is a custom feature in FAST v7, requiring a separate executable. In FAST v8.08, it is part of the standard distribution.

General

FAST Features	v7.02	v8.03
• Time marching	✓	✓
• Operating-point determination	✓	
• Linearization	✓	
• FAST-to-ADAMS preprocessor	✓	
• Follows the new FAST modularization framework		✓
• Structural and control routines separated from driver code		✓
• Independent time steps between modules	✓[†]	✓[‡]
• Independent spatial discretization between modules		✓
• Multiple integration options		✓
• Loose coupling with predictor-corrector across modules	✓[§]	✓
• Both 32-bit and 64-bit applications available	✓	✓[**]
• Supports both Windows and Linux operating systems	✓	✓
• Optimized for efficiency	✓	
• Supports mixed Fortran/C		✓
• Compiles with gfortran	✓	✓[††]

Major changes in FAST

v8.08.00c-bjj

- Coupling between ElastoDyn, SubDyn, and HydroDyn was added, allowing FAST to model fixed-bottom offshore turbines, including multi-member substructures (e.g., tripods and jackets).
- The module input-output solves have been enhanced; see the following paper for theoretical details: http://www.nrel.gov/docs/fy14osti/60742.pdf.
- The mesh mapping algorithms have been enhanced; see the following paper for theoretical details: http://www.nrel.gov/docs/fy14osti/60742.pdf.
- We now use LAPACK routines for solving linear systems, which has increased the speed of the simulations.
- The glue code allows the option for time-step subcycling. Modules can now choose to use *smaller* time steps than the glue code, as long as the module time step is an integer divisor of the glue-code time step. Note that we have found no cases where this option would be necessary.
- New modules for ice loading were added: IceFloe and IceDyn[‡‡].

[†] These steps must be integer multiples of the structural time step.
[‡] These steps must be integer divisors of the glue-code time step. Future versions will allow integer multiples of the glue-code time step as well.
[§] FAST v7 is limited to one correction step and this correction step only applies to some modules.
[**] The 64-bit version of FAST v8.08 does not contain the ability to use the MAP module.
[††] The FAST v8.08 source code can be compiled using gfortran, however the offshore cases do not run with this compiled executable. We are working to find the problem and fix it.

- Another module for mooring lines was added: FEAMooring[‡‡].
- The names of output files generated by both FAST and its modules have been standardized, see Figure 4. Files generated by FAST are named

  <RootName>.<ext>

  and files generated by FAST modules are named

  <RootName>.<ModName>.<ext>

  where <RootName> is the root name of the primary FAST input file (the file name, including path, without the extension), <ModName> is an abbreviation for the module generating the file, and <ext> is the file extension. File extensions currently are

Output file extension	File type
sum	Summary file
out	Time-marching tabular text output
outb	Time-marching tabular binary output
ech	Echo of input file (primarily for debugging)

- The "Time Ratio" displayed at the end of a simulation now includes only the CPU time *after* initialization. This ratio was changed to help users better predict the amount of time longer simulations will take (e.g., extrapolating the time a 1-hr simulation will take based on running a 1-min simulation).
- Information about the Jacobian and time steps was added to the FAST summary file.
- Bugs in handling errors were fixed. (In FAST v8.03.02b-bjj, InflowWind did not return all of its errors to the glue code, so it was using zero wind velocity when it went outside the turbulence grid.)
- FAST no longer allows the tower influence model, "NEWTOWER," to be used in AeroDyn on floating offshore turbines. This tower influence model assumes the tower does not move, which is a poor assumption for floating turbines.
- FAST will now abort if ElastoDyn's **TowerBsHt** value is negative for floating offshore systems.
- We fixed a bug in ElastoDyn (which is also present in FAST v7.02.00d) where the linear teeter-damper moment did not use **TeetDmpP.**
- We fixed a problem where the ElastoDyn Azimuth channel would be negative in rare cases.
- We fixed a problem with ElastoDyn's implementation of AM4, which incorrectly initialized the state history if corrections steps were taken.
- We no longer allow extrapolation orders of 0 in FAST v8.08.00c-bjj.
- We updated the DISCON*.DLL files used in the 5MW model certification tests. Previously, they did not work if users did not have Intel Visual Fortran installed on the computers they ran the simulations on.
- We fixed some bugs in the AeroDyn input files of the NREL 5-MW land-based turbine.
- We set the time steps of the floating offshore certification tests to be the same as they were in FAST v7.02.00d

[‡‡] IceDyn and FEAMooring have been added to FAST v8.08.00c-bjj, but they are not complete and have not been tested well.

- We added certification tests for the OC3 Monopile, OC3 Tripod, OC4 Jacket, and OC4-DeepCwind Semi-Submersible models.
- We have added some more utility files to the FAST archive, including:
    - PlotFASToutput.m, a MATLAB function that can plot some or all channels of one or more FAST time-series output files.
    - ReadSubDynSummary.m, a MATLAB function that can read the SubDyn summary file and put the data into a Matlab data-structure.
- We have updated the MAP_win32.dll file distributed with FAST so that it no longer depends on python being installed on the computer running FAST.
- We have added a 64-bit FAST executable to the archive, as well as a 64-bit version of DISCON_win64.DLL, and a “dummy” 64-bit version of MAP. This executable may be useful for running long simulations of large fixed-bottom offshore models (e.g., the OC4 Jacket); it *cannot* run any models that want to call the MAP module.

FAST v8.08.00c-bjj is compiled with the components listed in Table 2.

Table 2: Components in FAST v8.08.00c-bjj

Component	Version	
Modules		ModName (for output files)
ElastoDyn	v1.01.06b-bjj	ED
AeroDyn	v14.02.01c-bjj	AD
InflowWind	v2.00.01b-bjj	IfW
ServoDyn	v1.01.02a-bjj	SrvD
HydroDyn	v2.01.01c-gjh	HD
SubDyn	v1.01.00a-rrd	SD
MAP	v0.97.01a-mdm	MAP
FEAMooring	v1.00.00-yhb	FEAM
IceFloe	v1.00.00	IceF
IceDyn	v1.00.01-by	IceD
Other Components		
NWTC Subroutine Library	v2.03.03b-bjj	
FAST Registry[§§]	v2.03.01	
Third Party Content		
LAPACK	v3.3 as part of Intel® Math Kernel Library; (v3.5.0 compiled with gfortran)	
ScaLAPACK	2.0.2	
FFTPACK	v4.1	

[§§] The FAST Registry reads input files from each module to auto-generate the *_Types.f90 files required for the FAST framework.

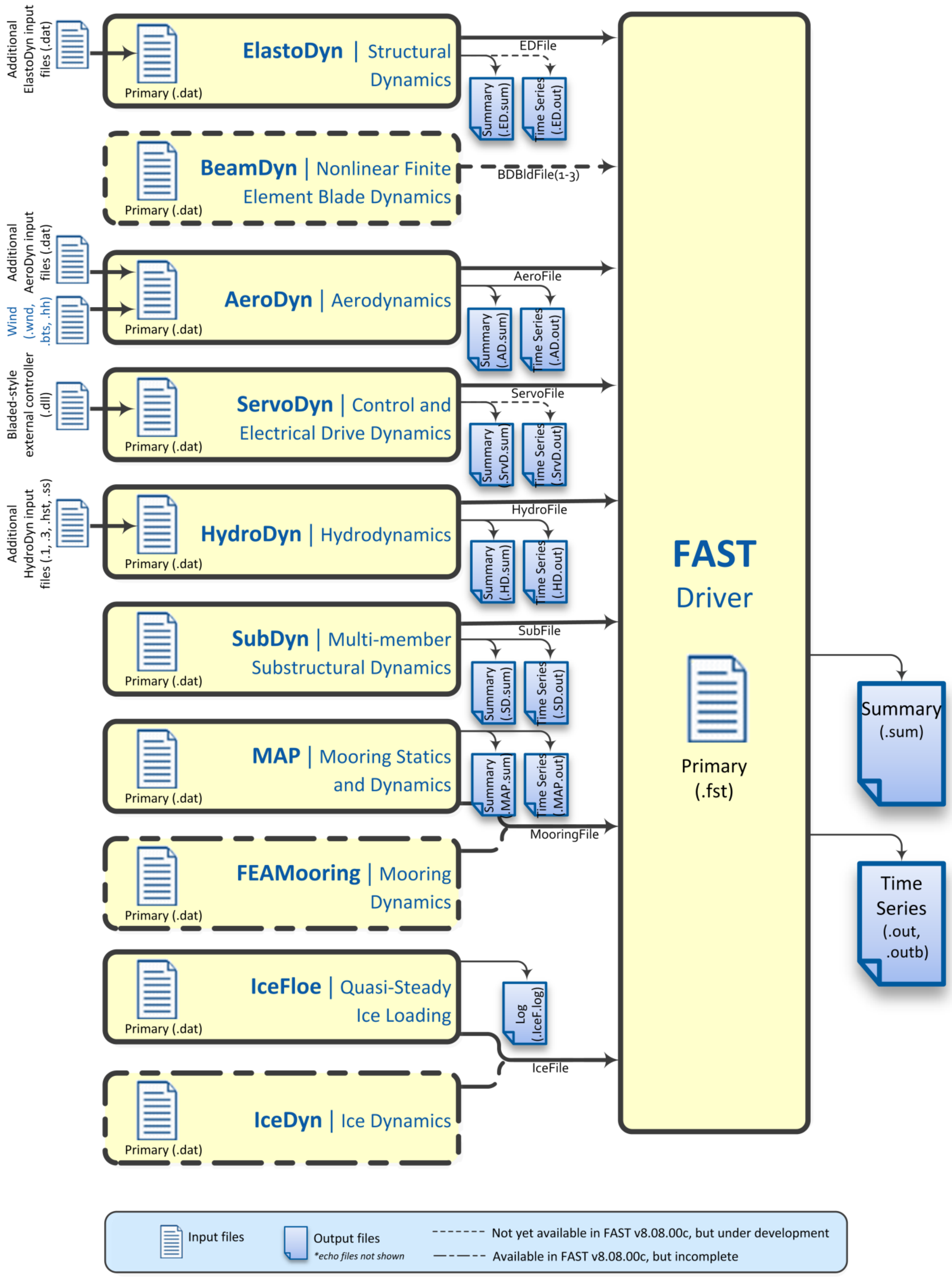


Figure 4: Summary of Input and Output Files for FAST v8.08.00c-bjj

v8.03.02b-bjj

Tasks completed to develop FAST v8 included:

- Converted FAST and its various modules (including AeroDyn and HydroDyn) into the new modularization framework (splitting out the controls and electrical-drive dynamics into a new ServoDyn module and structural dynamics into a new ElastoDyn module),
- Implemented a new driver program (glue code) supporting loose coupling of the modules,
- Developed mesh-to-mesh mapping schemes between module-independent discretizations of the spatial boundaries between modules,
- Coupled in the recently developed SubDyn module for multi-member substructure structural dynamics and MAP module for multi-segmented mooring quasi-statics, and
- Included a series of models using the NREL 5-MW Baseline wind turbine in the CertTest, including offshore configurations.

The driver program (glue code) couples the modules together; it controls the overall simulation progress and maps module outputs to inputs. We use the name “FAST” both for the driver program (glue code) and overall coupled code.

FAST and each of its modules, except InflowWind, have their own input files; see Figure 4.

Certification Tests

Several new models have been added to the FAST v8.08.00c-bjj archive. Table 3 lists the tests (1-25) and models available in the FAST CertTest folder:

Table 3: Certification Tests Distributed with FAST v8.08.00c-bjj

Test Name	Turbine Name	No. Blades (-)	Rotor Diameter (m)	Rated Power (kW)	Test Description
Test01	AWT-27CR2	2	27	175	Flexible, fixed yaw error, steady wind
Test02	AWT-27CR2	2	27	175	Flexible, steady wind
Test03	AWT-27CR2	2	27	175	Flexible, free yaw, steady wind
Test04	AWT-27CR2	2	27	175	Flexible, free yaw, turbulence
Test05	AWT-27CR2	2	27	175	Flexible, steady wind
Test06	AOC-15/50	3	15	50	Flexible, steady wind
Test07	AOC-15/50	3	15	50	Flexible, free yaw, turbulence
Test08	AOC-15/50	3	15	50	Flexible, fixed yaw error, steady wind
Test09	UAE VI downwind	2	10	20	Flexible, yaw ramp, steady wind
Test10	UAE VI upwind	2	10	20	Rigid, power curve, ramp wind
Test11	WP 1.5 MW	3	70	1500	Flexible, variable speed & pitch control, pitch failure, turbulence
Test12	WP 1.5 MW	3	70	1500	Flexible, variable speed & pitch control, ECD event
Test13	WP 1.5 MW	3	70	1500	Flexible, variable speed & pitch control, turbulence
Test14	*Not currently available*				
Test15	SWRT	3	5.8	10	Flexible, variable speed control, free yaw, EOG01 event
Test16	SWRT	3	5.8	10	Flexible, variable speed control, free yaw, EDC01

Test Name	Turbine Name	No. Blades (-)	Rotor Diameter (m)	Rated Power (kW)	Test Description
					event
Test17	SWRT	3	5.8	10	Flexible, variable speed control, free yaw, turbulence
Test18	NREL 5 MW - Land-based	3	126	5000	Flexible, DLL control, tower potential flow and drag, turbulence
Test19	NREL 5 MW - OC3-Monopile	3	126	5000	Flexible, DLL control, tower potential flow, turbulence, irregular waves
Test20	NREL 5 MW - OC3-Tripod	3	126	5000	Flexible, DLL control, tower potential flow, steady wind, regular waves with 0 phase
Test21	NREL 5 MW - OC4-Jacket	3	126	5000	Flexible, DLL control, tower potential flow, turbulence, irregular waves, marine growth
Test22	NREL 5 MW - ITI Barge	3	126	5000	Flexible, DLL control, irregular & multidirectional waves, turbulence
Test23	NREL 5 MW - MIT/NREL TLP	3	126	5000	Flexible, DLL control, turbulence, irregular waves
Test24	NREL 5 MW - OC3-Hywind	3	126	5000	Flexible, DLL control, turbulence, irregular waves
Test25	NREL 5 MW - OC4-DeepCwind Semi-Submersible	3	126	5000	Shortened OC4 Load Case (LC) 3.7: steady wind, white noise waves

Variables Specified in the FAST Primary Input File

FAST expects to find variables on specific lines in the text input file, so, do not add or remove lines in the file. The inputs listed in the file are described below, and an example file is provided at the end of this document, in Appendix: Example FAST v8.08.* Input File.

Simulation Control

This section of the input file contains options for controlling the simulation.

Echo: Echo input data to <RootName>.ech [T/F]

Setting this flag to “True” will result in the FAST primary input file being echoed to a file named “<RootName>.ech” where <RootName> is the name of the FAST primary input file, excluding its file extension. This feature is useful for debugging an input file.

AbortLevel: Error level when simulation should abort [“WARNING”, “SEVERE”, or “FATAL”]

This string tells FAST what error level should cause an abort. Typically we set this to abort on fatal errors, but there may be instances when a user wishes to abort on severe errors or warnings.

Fatal errors are those from which the program cannot recover. For example:

- Running out of memory when trying to allocate space for variables.
- Trying to read a number from a line of an input file that does not contain numeric values.
- Reaching the end of an input file before reading all the necessary information.
- Trying to open a file for writing, but the file is locked from another process.

Some examples of severe errors include the following:

- A format specifier for real numbers is too narrow to print "–1.0", so output files will almost certainly contain "***" instead of actual numbers.
- When trying to read a numeric value, logical "True" or "False" values were found instead. Fortran interprets them as 0 or 1, but that may not be what the user intended.
- A routine is using math based on the assumption that the angles are small, but the angles it found were larger than what it considers "small."

Warnings are typically generated when the simulation can continue—perhaps by the program adjusting inputs—but the results may not be what the user expected. Things that may generate warnings include

- Cases when user inputs are modified:
  - If the user asked for output on more tower strain gages than there are tower nodes, ElastoDyn will set the number of strain gages equal to the number of nodes.
  - If air density is set to zero, AeroDyn will turn off the dynamic-inflow model.
- Cases where non-physical conditions could be modeled:
  - If the user enables ElastoDyn's second flap mode but does not enable the first flap mode.
  - If the user has disabled wake calculations in AeroDyn.

TMax: Total run time [s]

This is the total length of the simulation to be run, in seconds. The first output is calculated at *t* = 0; the last output is calculated at *t* = **TMax** seconds.

DT: Recommended module time step [s]

This is the global, or glue-code, time step, **DT** is the value FAST will suggest modules use, although some modules may choose to use a time step that is an integer multiple smaller than **DT**. Module input-output relationships used to couple the modules together are calculated every **DT** seconds. It is essential that a small enough time step is used to ensure solution accuracy (by providing a sufficient sampling rate to characterize all key frequencies of the system), to ensure numerical stability of the selected time-integrators, and to ensure that the coupling between modules of FAST is numerically stable.

Our rule of thumb is to set **DT** = 1 / (10 * highest natural frequency in Hz in the coupled model). This natural frequency is hard to estimate before the full-system linearization of the coupled FAST model is realized. For coupled FAST models that don't use SubDyn, the frequency can be estimated via a linearization analysis of FAST v7. For coupled FAST models that do use SubDyn, guidance for choosing the time step is found in the SubDyn ReadMe file.

InterpOrder: Interpolation/Extrapolation order for input/output time history [1 or 2]

This is the order of the interpolation or extrapolation used for module inputs in the FAST glue code. Valid entries are "1" for linear interpolation/extrapolation or "2" for quadratic interpolation/extrapolation. Previous module inputs are extrapolated at the beginning of each step in the time-advancement loop to provide a guess for the actual module inputs at that step. Module inputs are typically interpolated in a module's UpdateStates routine.

We have found that quadratic extrapolation typically works well. However, there are times when linear extrapolation provides a stable solution while quadratic does not. We have found this to be true for cases where the model has poor initial values or cases where the simulation may have errors building up.

NumCrctn: Number of correction iterations [-]

This is the number of corrections to be taken on each step of the predictor-corrector scheme implemented in FAST. The value of **NumCrctn** must not be negative. Most models can achieve stable solutions by using explicit calculations (i.e., no corrections: **NumCrctn** = 0), particularly if using **InterpOrder** = 2 and the recommended **DT**—see above. However, corrections may be needed if you wish to achieve a given convergence rate of an underlying time integrator (e.g., if you are using a 4th-order accurate integration scheme, you may only get a 2nd-order accurate solution with no corrections. If you desire a 4th-order accurate solution, you may need one or more corrections).

DT_UJac: Time between calls to get Jacobians [s]

We use a Jacobian matrix to solve the module input-output relationship between accelerations and loads in the ElastoDyn-HydroDyn-SubDyn coupling. This Jacobian is computed with finite differences and can be time consuming. However, it rarely needs to be calculated frequently.

DT_UJac determines how often the Jacobian needs to be updated. If the platform reference point in ElastoDyn doesn't rotate much, **DT_UJac** can be set to a value larger than **TMax**. **DT_UJac** is not currently used for land-based systems. For floating systems, where the platform may rotate more than several degrees in roll, pitch, and/or yaw, it is recommend to set **DT_UJac** = 1/(10*natural frequency in Hz of the roll, pitch, or yaw mode with excessive motion).

UJacSclFact: Scaling factor used in Jacobians [-]

This factor is used to divide the magnitude of the load terms in the Jacobian (see **DT_UJac)** so that they are approximately the same order of magnitude as the acceleration terms. For the NREL 5-MW turbine models in the Certification Test, we've set it to 1E+06 and have not found any cases where that value did not work. **UJacSclFact** may need to be set larger or smaller when modeling wind turbines much larger or smaller turbines than the NREL 5-MW baseline.

Feature Switches and Flags

This section of the input file contains switches and flags that tell FAST which modules should be used in the simulation.

CompElast: Compute structural dynamics [1 or 2]

1: Use ElastoDyn for the structural dynamics of the rotor, drivetrain, nacelle, and tower
2: Use BeamDyn for the structural dynamics on the blades and ElastoDyn for the drivetrain, nacelle, and tower. ***CompElast** = 2 is currently disabled; it will be implemented when BeamDyn is integrated into FAST.*

Please note that ElastoDyn must always be used when running FAST.

CompAero: Compute aerodynamic loads [0 or 1]
0: Do not calculate aerodynamic loads

1: Use AeroDyn for aerodynamic loads

CompServo: Compute control and electrical-drive dynamics [0 or 1]
0: Do not calculate control and electrical-drive dynamics

1: Use ServoDyn for control and electrical-drive dynamics

CompHydro: Compute hydrodynamic loads [0 or 1]
0: Do not calculate hydrodynamic loads

1: Use HydroDyn for hydrodynamic loads

If **CompHydro** is not zero, FAST considers the model to be an offshore system. If **CompSub** is also non-zero, the offshore system is a fixed-bottom system. If **CompSub** is zero, the offshore system is considered a floating system.

CompSub: Compute sub-structural dynamics [0 or 1]
0: Do not calculate sub-structural dynamics

1: Use SubDyn for sub-structural dynamics

CompMooring: Compute mooring system [0, 1, or 2]
0: Do not model a mooring system

1: Use MAP to model a mooring system

2: Use FEAMooring to model a mooring system

Note that FEAMooring is not complete (not fully functional) in FAST v8.08.00c-bjj.

CompIce: Compute ice loads [0, 1, or 2]
0: Do not model offshore surface ice

1: Use IceFloe to model offshore surface ice

2: Use IceDyn to model offshore surface ice

Note that IceDyn is not complete (not fully functional) in FAST v8.08.00c-bjj.

CompUserPtfmLd: Compute additional platform loading [T/F]
This feature is currently disabled.

CompUserTwrLd: Compute additional tower loading [T/F]
This feature is currently disabled.

Input Files
The input files specified in this section can be specified relative to the location of the FAST primary input file or specified with an absolute path. We recommend that you use quotes around the path/file names.

EDFile: Name of file containing ElastoDyn input parameters [-]
This is the name of the ElastoDyn primary input file.

BDBldFile(1): Name of file containing BeamDyn input parameters for blade 1 [-]
This feature is currently disabled.

BDBldFile(2): Name of file containing BeamDyn input parameters for blade 2 [-]
This feature is currently disabled.

BDBldFile(3): Name of file containing BeamDyn input parameters for blade 3 [-]
This feature is currently disabled.

AeroFile: Name of file containing aerodynamic input parameters [-]
This is the name of the AeroDyn primary input file. At this time, **AeroFile** must be specified *even if* ***CompAero*** *= 0* because ElastoDyn currently reads the AeroDyn input file for its blade discretization. We plan to fix this to use the mesh-mapping required in the FAST framework in a future release.

ServoFile: Name of file containing control and electrical-drive input parameters [-]
This is the name of the ServoDyn primary input file. It is not used if **CompServo** = 0.

HydroFile: Name of file containing hydrodynamic input parameters [-]
This is the name of the HydroDyn primary input file. It is not used if **CompHydro** = 0.

SubFile: Name of file containing sub-structural input parameters [-]
This is the name of the SubDyn primary input file. It is not used if **CompSub** = 0.

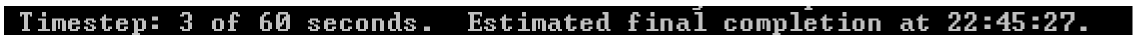
MooringFile: Name of file containing mooring system input parameters [-]
This is the name of the MAP (**CompMooring** = 1) or FEAMooring (**CompMooring** = 2) primary input file. It is not used if **CompMooring =** 0.

IceFile: Name of file containing ice input parameters [-]
This is the name of the IceFloe (**CompIce** = 1) or IceDyn (**CompIce** = 2) primary input file. It is not used if **CompIce =** 0.

Output
This section deals with what can be output from a FAST simulation.

SumPrint: Print summary data to "<RootName>.sum" [T/F]
When set to "true", FAST will generate a file named "<RootName>.sum". This summary file contains the version number of all modules being used, the time steps for each module, and information about the channels being written to the time-marching output file(s). If **SumPrint** is "false", no summary file will be generated.

SttsTime: Amount of time between screen status messages [s]
During a FAST simulation, the program prints a message like this:



```
 Timestep: 3 of 60 seconds.  Estimated final completion at 22:45:27.
```

SttsTime sets how frequently this message is updated. For example, if **SttsTime** is 2 seconds, you will see this message updated every 2 seconds of *simulation* time.

DT_Out: Time step for tabular output [s]
This is the time step of the data in the tabular (time-marching) output files. **DT_Out** must be an integer multiple of **DT.**

TStart: Time to begin tabular output [s]
This is time step that must be reached before FAST will begin writing data in the tabular (time-marching) output files. Note that the output files may not actually start at **TStart** seconds if **TStart** is not an integer multiple of **DT_Out.**

OutFileFmt: Format for tabular output [1, 2, or 3]
This indicates which type of tabular (time-marching) output files will be generated. If **OutFileFmt** is 1, only a text file will be written. If **OutFileFmt** is 2, only a binary file will be written. If **OutFileFmt** is 3, both text and binary files will be written.

Text files write a line to the file each time step. This can make the simulation run slower, but it can be useful for debugging, particularly if a simulation doesn't run to completion or if you want to look at some results before the entire simulation finishes.

Binary files write the entire file at the end of the simulation. If a lot of output channels are requested for a long simulation, this can take up a moderate amount of memory. However, they tend to run faster and the resulting files take up much less space. The binary files contain more precise output data than text files, which are limited by the chosen output format specifier—see **OutFmt** below.

We recommend you use text files for debugging and binary files for production work. A MATLAB script for reading FAST binary output files is included in the archive (see Utilities\SimulationToolbox\Utilities\ReadFASTbinary.m). The NREL post-processors Crunch and MCrunch can also read these binary files.

TabDelim: Use tab delimiters in text tabular output file? [T/F]
When **OutFileFmt** = 1 or 3, setting **TabDelim** to "True" will put tabs between columns in the text tabular output file. Otherwise, spaces will separate columns in the text tabular output file. If **OutFileFmt** = 2, **TabDelim** has no effect.

OutFmt: Format used for text tabular output, excluding the time channel [-]
When **OutFileFmt** = 1 or 3, FAST will use **OutFmt** to format the channels printed in the text tabular output file. **OutFmt** should result in a field that is 10 characters long (channel headers are 10 characters long, and NWTC post-processing software sometimes assume 10 characters). The time channel is printed using the "F10.4" format. We commonly specify **OutFmt** to be "ES10.3E2".

If **OutFileFmt** = 2, **OutFmt** has no effect.

Modeling Tips

If a model is numerically unstable, you can try these steps

- Add a correction step (**NumCrctn**).
- Make **DT** smaller.
- Change **InterpOrder**.
- Set better initial conditions in the module input files (particularly ElastoDyn).

Limitations

Table 1 shows a comparison of features between FAST v7 and FAST v8.08.00c-bjj. It is our intent to include all of the features of FAST v7 in the new modular framework of FAST v8 in the future. Until then, NREL plans to support both versions of FAST. Other limitations of FAST v8 include:

- FAST v8.08.00c-bjj is distributed as both a 32-bit and 64-bit Windows application. However, the 64-bit application cannot run the MAP module. We are working to update the MAP DLL so that it does not depend on third-party libraries that are not supported in 64-bit Windows applications.
- FAST v8.08. 00c-bjj runs a bit slower than FAST v7.02.00d-bjj (though the offshore cases run significantly faster than FAST v8.03.02b-bjj). We have put our effort into getting the framework to work and will address computational efficiency later. We expect great improvements in efficiency as development continues.
- The OC4 Jacket model requires approximately 2 GB of memory and may not run on 32-bit Windows® systems. (The model does run using FAST_Win32.exe on a 64-bit Windows® system.)
- The IceDyn and FEAMooring modules are not complete in FAST v8.08.00c-bjj.

Future Work

All future developments of FAST will follow the framework.

- Address items from the “Limitations” section
- Introduce the new BeamDyn module for nonlinear finite-element modeling of blade structural dynamics
- Upgrade the loose-coupling algorithm of the glue code to allow each module to have its own time step, including time steps larger than the glue-code time step.
- Upgrade ElastoDyn and AeroDyn to have distinct discretization schemes for the blades and tower (currently ElastoDyn uses AeroDyn’s blade discretization and AeroDyn uses ElastoDyn’s tower discretization)
- Optimize the code, particularly ElastoDyn, so that it runs faster
- Introduce tight coupling

- Introduce operating-point determination and linearization across the coupled aero-hydro-servo-elastic solution
- And much, much, more…

Converting to FAST v8.08.x

We have created template input files for FAST v8.08.x, ElastoDyn v1.01.x, ServoDyn v1.01.x, and HydroDyn v2.01.x. These template files can be found in the Matlab Simulation Toolbox that is now included in the FAST archive: Utilities\SimulationToolbox\ConvertFASTVersions\TemplateFiles.

See the "Matlab Conversion Scripts" section below for help in automatically converting input files to the latest version.

Also note that you can find example up-to-date input files in the FAST v8.08.00c-bjj archive's CertTest folder. See the "Certification Tests" section of this document for descriptions.

Summary of Changes to Inputs

This section summarizes changes to the FAST primary input file between major releases.

Changes in FAST v8.08.00c-bjj

The following list describes the differences in the primary input file of FAST v8.08.00c-bjj relative to FAST v8.03.02b-bjj.

- Many variables in the primary FAST input file have been renamed:

FAST v8.03.x Name	FAST v8.08.x Name
ADFile	AeroFile
SrvDFile	ServoFile
HDFile	HydroFile
SDFile	SubFile
MAPFile	MooringFile
CompMAP	CompMooring

- Most of the feature *flags* were changed to feature *switches* in the primary FAST input file. Instead of True/False inputs, CompAero, CompServo, CompHydro, CompSub, and CompMooring now require integer inputs.
- Inputs for coupling with modules IceFloe and IceDyn have been added: **CompIce** and **IceFile**.
- Several new inputs for future coupling of BeamDyn into FAST have been added. These inputs are **CompElast**, **BDBldFile(1)**, **BDBldFile(2)**, and **BDBldFile(3)**.
- The modules ElastoDyn, ServoDyn, AeroDyn, HydroDyn, and SubDyn allow users to input the string "Default" for their respective time steps, which will then use the time step from the FAST primary input file.

Changes in FAST v8.03.02b-bjj

The following list describes the differences in the FAST, ElastoDyn, and ServoDyn input files of FAST v8.03.02b-bjj relative to the input files of FAST v7.02.00d-bjj.

- The primary FAST input file has been converted to primary input files for FAST, ElastoDyn, and ServoDyn and some of the inputs have been reordered.
- The FAST Platform file has been eliminated, with some of the inputs now part of the ElastoDyn primary input file and some of the inputs now part of HydroDyn's and MAP's input files.
- All of the inputs formerly labeled "[CURRENTLY IGNORED]" have been removed.
- Switches for ADAMS preprocessing and linearization have been removed.
- Noise has been removed.
- **PtfmLdMod** has been converted to **CompUsrPtfmLd**.
- **TwrLdMod** has been converted to **CompUserTwrLd**.
- The tip-brake inputs have been removed.
- **PtfmCM** is now **PtfmCMzt**, with **PtfmCMzt** = -**PtfmCM**.
- Corresponding inputs **PtfmCMxt** and **PtfmCMyt** have been added.
- **PtfmRef** is now **PtfmRefzt**, with **PtfmRefzt** = -**PtfmRef**.
- **TwrRBHt** and **TwrDraft** have been replaced with **TowerBsHt**, with **TowerBsHt** = **TwrRBHt** – **TwrDraft**.
- The output decimation factor (**DecFact**) has been converted to **DT_out** (**DT_out** = **DT*****DecFact**).
- The yaw and pitch maneuvers no longer specify end times for the maneuvers. Instead they specify a rate for the maneuver.
- The **GBRevers** variable has been removed; input **GBRatio** must now be specified as a negative number if **GBRevers** was previously set to True.
- ElastoDyn's blade input properties table no longer specifies **AeroCent**. Instead, it specifies the location of the pitch axis, **PitchAxis**, which is calculated as **PitchAxis** = 0.5 – **AeroCent** by the MATLAB conversion script; the aerodynamic center will become part of AeroDyn in a future release.
- The **OutList** variables have been divided among the various FAST modules, and several outputs are no longer valid.
- The GH Bladed Interface is now a standard option in ServoDyn, without requiring a recompile.
- Tower drag loading has been added to AeroDyn v14.02.00c-mlb with a new corresponding flag in the AeroDyn input file.
- The glue code allows options for **AbortErrLevel**, number of corrections in the predictor-corrector algorithm, and extrapolation/interpolation order of module inputs to be used for time advancement.
- For the ElastoDyn coupling to HydroDyn or SubDyn, FAST also has two inputs controlling the implicit solve (via Jacobian computed with finite differences)—see **DT_UJac** and **UJacSclFact** described in the Variables Specified in the FAST Primary Input File Section above.

Matlab Conversion Scripts

Because the changes to the input files are significant, we have created Matlab scripts to automatically convert FAST v7.x or FAST 8.03.02b-bjj input files to FAST v8.08.00c-bjj. The files you will need are included in the Simulation Toolbox, located in this directory of the FAST archive: Utilities\SimulationToolbox\ConvertFASTVersions.

We recommend that you add the Simulation Toolbox to your Matlab path so that you can access all of the routines defined in it. For example:

```
FASTSimulationToolbox = 'C:\Users\bjonkman\FAST\UtilityCodes\SimulationToolbox';
addpath( genpath( FASTSimulationToolbox ) );
```

An example showing how we converted the NREL CertTest input files for use with FAST v8.08.00c-bjj is included in the FAST archive: CertTest\ConvertFiles.m. You can use this script as a basis for helping to convert your own input files; however, we *strongly* recommend that you make copies of all your input files before running any scripts to convert them. Fortran and MATLAB read text input files in slightly different ways; so some things that may work in Fortran may not be read in the same way in MATLAB. We recommend that you carefully check the files after converting them.

Converting from FAST v8.03.02b-bjj

If you have FAST v8.03.x input files that you wish to convert to FAST v8.08.00c-bjj, you can use the Matlab function, **ConvertFAST8_3to8**. This function will convert the primary FAST input file *and the primary HydroDyn input file* from FAST v8.03.x to FAST v8.08.x. You will need to provide the name of the FAST v8.03.x primary input file and the name of the directory where the new input files should be placed:

```
ConvertFAST8_3to8( inputfile, newDir );
```

Note that we do not automatically convert the SubDyn input files from FAST v8.03. These you must do by hand—or use the models provided in the FAST Certification Tests.

Converting from FAST v7

To convert FAST v7 input files to FAST v8.08.00c-bjj, you can use the MATLAB function **ConvertFAST7to8**. This function will create a new primary input files for FAST, ServoDyn, and ElastoDyn. It also creates new blade and tower files for ElastoDyn. *It does not automatically convert hydrodynamic- and mooring system-related inputs*. This function does not create HydroDyn, SubDyn, MAP, or AeroDyn[***] input files.

You will need to provide the **ConvertFAST7to8** function with the name of the FAST v7.x primary input file and the directory where the new input files should be placed. *The new directory should not be the directory where the old files are located!*

```
ConvertFAST7to8(inputfile,newDir);
```

[***] If you use the "NEWTOWER" model in AeroDyn, you will have to add one additional line (the AeroDyn input file for Test18 in the CertTest folder contains this line); if you do not use the "NEWTOWER" model, the AeroDyn input file has not changed.

If your input file has pitch or yaw maneuvers, you may also provide the routine with the new rates (instead of the end times previously used). We have also provided a Matlab routine (**CalculateYawAndPitchRates**) that will calculate these rates, but you must provide the routine the name of the FAST output file that contains the previous results of the Pitch and/or Yaw channels.

```
[YawManRat, PitManRat] = CalculateYawAndPitchRates('Test09.fst', 'Test09.out');
ConvertFAST7to8(inputfile, newDir, YawManRat, PitManRat);
```

If **YawManRat** or **PitManRat** are zero, those inputs are ignored and values from the ServoDyn template file will be used instead.

If your input file was used with the custom interface to GH Bladed DLL controllers, you should also set the optional input parameter, usedBladedDLL, so that your input switches that previously called the DLL are now set to 5, the new switch for User-Defined Control from Bladed DLL.

```
ConvertFAST7to8(oldFSTName, newDir, YawManRat, PitManRat, usedBladedDLL)
```

Compiling

If you want to compile the code, please read the file, “CompilingInstructions_FASTv8.pdf” located in the “Compiling” folder in the FAST archive for details.

Unlike FAST v7 distributions, all of the source code you need to compile the project is contained in the archive’s Source directory. The text files (.txt) in the source folders are input files for the FAST Registry. These files are used to generate the *_Types.f90 files for the component modules.

The compiling folder contains:

- A Microsoft Visual Studio project with all of the FAST source files and settings needed to compile in Release or Debug mode. This project places executables called “FAST_dev_Win32.exe” and “FAST_dev_debug_Win32.exe” in the FAST\bin folder. (renamed with “_dev” to prevent overwriting the executables created by NREL and distributed with FAST).
- A Windows® batch script that can be run from your Intel Fortran Command Prompt Window, with very little (if any) modification. This batch file creates executables named “FAST_iwin32.exe” and “FAST_iwin64.exe” in the local (i.e., FAST\Compiling) folder.
- A *makefile* for gfortran with (most of) the settings to create “FAST_gwin32.exe” and “FAST_gwin64.exe” in the local (i.e., FAST\Compiling) folder. To run this on Windows, you will need to install binaries of the LAPACK libraries. Please see “CompilingInstructions_FASTv8.pdf” for details. This *makefile* has been tested only on Windows. Also note that offshore models do not run with the gfortran executables (land-based models do).

All of these tools for compiling are set up to compile and link with the appropriate settings, though you may have to modify the *makefile* to find the LAPACK libraries. (Note that FAST uses several specialized compiling/linking options and that MAP is distributed as a dynamic-link library.) These tools also run the FAST Registry if necessary (e.g., changes to the Registry input files or missing *_Types.f90 files).

The Visual Studio project and Windows batch script were created using

Microsoft Visual Studio 2010 Shell
Intel® Visual Fortran Composer XE 2011, version 12.1.3.300
Intel Math Kernel Library (MKL) 10.3 Update 9

The MKL is used only for LAPACK routines.

Running FAST v8.08.00c-bjj

FAST v8.08.00c-bjj must load the MAP library when the program starts. FAST_Win32.exe needs to load MAP_Win32.dll and FAST_x64.exe needs to load MAP_x64.dll. These dynamic link libraries (DLLs) must be on your Windows® path. The easiest way to do this is to make sure that the MAP DLLs are in the same directory as the FAST executables. We distribute the executables and DLLs in the \bin directory of the FAST archive, so this is already done for you. However, if you choose to move the files or if you compile using the Windows® batch script or the *makefile* for gfortran, you may have to modify your path environment variable or move some files.

To run FAST from a Windows command prompt, the syntax is:

```
<name of FAST executable with or without extension> <name of input file with extension>
```

To start, it easiest to open up your command window in the directory in which your FAST primary input file and FAST executable are stored. For example, if you have an input file named "Input.fst", along with "FAST_Win32.exe", stored in "C:\FileLocation", you should type:

```
C:\>cd FileLocation
C:\FileLocation> FAST_Win32 Input.fst
```

The syntax is the same for different input files. Simply change "Input.fst" to whatever input file you want.

An installation guide is available that describes how to install FAST (and the other CAE tools) in such a way that they will run from a command window from any folder (without moving or copying the executable around to different folders). See: http://wind.nrel.gov/designcodes/papers/setup.pdf.

Feedback

If you have questions or wish to provide feedback, please use our forums:
https://wind.nrel.gov/forum/wind/

Appendix: Example FAST v8.08.* Input File

```
------- FAST v8.08.* INPUT FILE ------------------------------------------------
NREL 5.0 MW Baseline Wind Turbine with OC4 Jacket Configuration, for use in offshore analysis
---------------------- SIMULATION CONTROL --------------------------------------
False      Echo            - Echo input data to <RootName>.ech (flag)
"FATAL"    AbortLevel      - Error level when simulation should abort (string) {"WARNING", "SEVERE", "FATAL"}
        60   TMax            - Total run time (s)
      0.01 DT              - Recommended module time step (s)
         2 InterpOrder     - Interpolation order for input/output time history (-) {1=linear, 2=quadratic}
         1 NumCrctn        - Number of correction iterations (-) {0=explicit calculation, i.e., no corrections}
     99999 DT_UJac         - Time between calls to get Jacobians (s)
     1E+06 UJacSclFact     - Scaling factor used in Jacobians (-)
---------------------- FEATURE SWITCHES AND FLAGS ------------------------------
         1 CompElast       - Compute structural dynamics (switch) {1=ElastoDyn; 2=ElastoDyn+BeamDyn blades}
         1 CompAero        - Compute aerodynamic loads (switch) {0=None; 1=AeroDyn}
         1 CompServo       - Compute control and electrical-drive dynamics (switch) {0=None; 1=ServoDyn}
         1 CompHydro       - Compute hydrodynamic loads (switch) {0=None; 1=HydroDyn}
         1 CompSub         - Compute sub-structural dynamics (switch) {0=None; 1=SubDyn}
         0 CompMooring     - Compute mooring system (switch) {0=None; 1=MAP; 2=FEAMooring}
         0 CompIce         - Compute ice loads (switch) {0=None; 1=IceFloe; 2=IceDyn}
False      CompUserPtfmLd - Compute additional platform loading (flag) {false: none; true: user-defined}
False      CompUserTwrLd  - Compute additional tower loading (flag) {false: none; true: user-defined}
---------------------- INPUT FILES ---------------------------------------------
"OC4Jacket_ElastoDyn.dat" EDFile       - Name of file containing ElastoDyn input parameters (quoted string)
"unused"                  BDBldFile(1) - Name of file containing BeamDyn blade 1 inputs (quoted string)
"unused"                  BDBldFile(2) - Name of file containing BeamDyn blade 2 inputs (quoted string)
"unused"                  BDBldFile(3) - Name of file containing BeamDyn blade 3 inputs (quoted string)
"OC4Jacket_AeroDyn.dat"   AeroFile     - Name of file containing aerodynamic input parameters (quoted string)
"OC4Jacket_ServoDyn.dat"  ServoFile    - Name of file with control/electric-drive inputs (quoted string)
"OC4Jacket_HydroDyn.dat"  HydroFile    - Name of file containing hydrodynamic inputs (quoted string)
"OC4Jacket_SubDyn.dat"    SubFile      - Name of file containing sub-structural inputs (quoted string)
"unused"                  MooringFile  - Name of file containing mooring system inputs (quoted string)
"unused"                  IceFile      - Name of file containing ice input parameters (quoted string)
---------------------- OUTPUT --------------------------------------------------
True       SumPrint        - Print summary data to "<RootName>.sum" (flag)
         1 SttsTime        - Amount of time between screen status messages (s)
      0.05 DT_Out          - Time step for tabular output (s)
         0 TStart          - Time to begin tabular output (s)
         2 OutFileFmt      - Format for tabular (time-marching) output file (switch) {1:out, 2:outb, 3:both}
True       TabDelim        - Use tab delimiters in text tabular output file? (flag)
"ES10.3E2" OutFmt          - Format used for text tabular output, excluding the time channel. (quoted string)
```

Figure 5: Example FAST v8.08.00c-bjj Input File